# NAVAL POSTGRADUATE SCHOOL
## Monterey, California


# THESIS

**EVALUATION OF THE EXTENSIBLE MARKUP LANGUAGE (XML) AS A MEANS OF ESTABLISHING INTEROPERABILITY BETWEEN MULTIPLE DEPARTMENT OF DEFENSE (DOD) DATABASES**

by

Eddie L. Davis

June 2001

| | |
|---|---|
| Thesis Advisor: | Valdis Berzins |
| Second Reader: | CAPT Paul Young |

**Approved for public release; distribution is unlimited.**

# REPORT DOCUMENTATION PAGE

*Form Approved*
*OMB No. 0704-0188*



| 1. AGENCY USE ONLY *(Leave blank)* | 2. REPORT DATE | 3. REPORT TYPE AND DATES COVERED |
|---|---|---|
| | June 2001 | Master's Thesis |

**4. TITLE AND SUBTITLE**
EVALUATION OF THE EXTENSIBLE MARKUP LANGUAGE (XML) AS A MEANS FOR ESTABLISHING INTEROPERABILITY BETWEEN HETEROGENEOUS DEPARTMENT OF DEFENSE (DOD) DATABASES

**5. FUNDING NUMBERS**

**6. AUTHOR(S)**
Davis, Eddie L.

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**
Naval Postgraduate School
Monterey, CA 93943-5000

**8. PERFORMING ORGANIZATION REPORT NUMBER**

**9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)**

**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**

**11. SUPPLEMENTARY NOTES**
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

**12a. DISTRIBUTION / AVAILABILITY STATEMENT**
Approved for public release; distribution is unlimited.

**12b. DISTRIBUTION CODE**

**13. ABSTRACT *(maximum 200 words)***

This thesis evaluates the ability of the Extensible Markup Language (XML) to address the interoperability problem that exists between Department Of Defense (DOD) legacy systems. Due to the different Database Management Systems (DBMS) used within DOD, interoperability is a major flaw. The need for communication between the DBMS is necessary and this thesis will focus on this problem.

This thesis focuses in on the problems that exist, and assesses XML as a means of correcting these problems. This thesis uses the Joint Common Database (JCDB) as a means of showing XML to be a viable solution.

**14. SUBJECT TERMS**
Extensible Markup Language, Interoperability, Database Management

**15. NUMBER OF PAGES**
83

**16. PRICE CODE**

| 17. SECURITY CLASSIFICATION OF REPORT | 18. SECURITY CLASSIFICATION OF THIS PAGE | 19. SECURITY CLASSIFI- CATION OF ABSTRACT | 20. LIMITATION OF ABSTRACT |
|---|---|---|---|
| Unclassified | Unclassified | Unclassified | UL |

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)

THIS PAGE INTENTIONALLY LEFT BLANK

**Approved for public release; distribution is unlimited**

# EVALUATION OF THE EXTENSIBLE MARKUP LANGUAGE (XML) AS A MEANS FOR ESTABLISHING INTEROPERABILITY BETWEEN MULTIPLE DEPARTMENT OF DEFENSE (DOD) DATABASES

Eddie L. Davis
Mississippi Valley State University, 1984

Submitted in partial fulfillment of the
requirements for the degree of

**MASTER OF SCIENCE IN SOFTWARE ENGINEERING**

from the

**NAVAL POSTGRADUATE SCHOOL**
**June 2001**

Author: Eddie L. Davis

Eddie L. Davis

Approved by: Valdis Berzins

Valdis Berzins, Thesis Advisor

Paul Young

Paul Young, Second Reader

Luqi, Chairman
Software Engineering Curriculum

THIS PAGE INTENTIONALLY LEFT BLANK

# Abstract

This thesis evaluates the ability of the Extensible Markup Language (XML) to address the interoperability problem that exists between Department Of Defense (DOD) legacy systems. Due to the different Database Management Systems (DBMS) used within DOD, interoperability is a major flaw. The need for communication between the DBMS within DOD is necessary and this thesis will focus on this problem.

This thesis focuses in on the problems that exist, and assesses XML as a means of correcting these problems. This thesis uses the Joint Common Database (JCDB) as a means of showing XML to be a viable solution.

THIS PAGE INTENTIONALLY LEFT BLANK

# TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

## ACKNOWLEDGEMENTS

The author wishes to thank Dr. Valdis Berzins for his guidance and support. A special thanks go out to the TACOM Software Engineering Masters of Science students for their support.

# I. INTRODUCTION

## A. SUMMARY

The Joint Battlefield Infosphere (JBI) has introduced the concept of sharing data across different domains. The question of how and what to share has to be addressed. The what is DOD legacy databases. These databases have been in existence for many years. At the time of their creation, these systems didn't address the concept of sharing data. Therefore, in order to make this idea of sharing come to fruition, the gap must be bridged. This thesis focuses in on XML as being that bridge. If so, this would address how sharing can take place.

Though this concept seems to be straightforward, it isn't. The DOD databases have multiple formats. To find a bridge that would cover sharing between all current formats may be a little harder to accomplish. This is mainly because the sharing has to be both ways. There must be a means of getting the data across the bridge and back. It may be straight forward for the bridge to accept the data but to put it in a specific format for return may pose a problem in some instances.

Based on the discussion above, XML has a good potential of solving most if not all the issues discussed with the help of some Commercial Off The Shelf (COTS) Tools.

This thesis focuses on the Joint Common Database (JCDB) as a means of showing that XML could solve the interoperability issue with a portion of the legacy systems within DOD. Since JCDB was designed in Informix, the focus of discussion will be on what it will take to make XML a tool for data exchange in this context.

## B. RESEARCH QUESTIONS

This thesis focuses on providing answers to the following questions:

1. What is required in order to exchange data?
2. What XML COTS Tools are available to provide this data exchange?
3. Will these XML COTS Tools provide a complete solution?
4. Are there other potential solutions?

These questions are addressed here in the context of the Joint Common Database (JCDB). The JCDB is just one of the DOD legacy systems. It is being used as an illustration throughout this thesis.

## C. MOTIVATION

In software development at the present time, developers attempt to make systems as open as possible. This idea would allow the system to be extended/enhanced later with less effort on the developer and less effect on the current configuration. Unfortunately, this was not the case when the DOD legacy systems were developed. This thesis will provide an analysis in the area of interoperability using XML.

Once the idea of interoperability is discussed and solutions proven, this could be a big step for DOD. Interoperability seems to always be something that is an afterthought during the initial development of a DOD system. This concept has to be brought to the forefront when developing hardware as well as software systems. It is too costly to attempt to correct the problem after development has taken place. Hopefully some ideas

in this thesis will raise the awareness of developers to the importance of considering interoperability, upfront and early.

## D. ORGANIZATION

This thesis provides information in the following manner:

1. Chapter II provides background information about the DOD interoperability issues. It then provides what is considered a potential solution. Background information on the proposed solution is also presented. Finally, information on the capabilities of XML and how XML may be used to solve the problem of interoperability are discussed.
2. Chapter III gives a brief description of the databases that are of interest for DOD. It also gives insight on the Database Management Systems (DBMS) that make up the DOD legacy systems.
3. Chapter IV provides information on the findings during the analysis of XML to solve the data interoperability issue. It also provides other alternatives that may be considered as a means of alleviating the data interoperability problem between the DOD legacy system.
4. Chapter V provides the conclusions and recommendations that were arrived at by the author.

THIS PAGE INTENTIONALLY LEFT BLANK

# II. BACKGROUND

## A. DOD INTEROPERABILITY ISSUES

### 1. Understanding The Issues

DOD has jumped on the bandwagon of digital information. Because of the widespread spectrum of the information needed at the DOD level, there is a big diversity in the type of information needed and/or used.

DOD is broken into different departments. Though they are part of the whole DOD, they still operate as separate entities. This has its advantages and disadvantages in respect to digitized information. One main advantage is, by acting as a separate entity the end product may be completed faster. By the same token, without the right guidelines every department will accomplish the tasks in different fashions. This could be a big disadvantage in the long run.

In the past this is how DOD has done it. Many legacy systems were developed with no thought of what's to come. The idea of open-ended architecture was not a consideration. The only thought has been of doing the minimum that needs to be done to provide an operational product.

This was common in the past. Now with the concept/idea of a digital battlefield, there is a need for communication between departments within DOD. Unfortunately, to no fault of their own, DOD has many different format/ configurations that have to be bridged together in order for interoperability to be accomplished. This can only be accomplished if the well-documented barriers for data interoperability within DOD are addressed. [NRC]

## 2. DOD Barriers to Data Interoperability

### *a. Time and Funds*

DOD has built systems that have been in existence for many years. Changing these systems to be interoperable at this point is not feasible. This task would be time consuming not to mention extremely expensive.

Though the systems may be old they are currently being used and therefore can't be abandoned. Therefore, some phased approach has to be arrived at to accomplish interoperability.

### *b. Close Ended Design*

The legacy systems in question were designed during a time when the main objective was to develop a system that would provide what was needed at that time. The end result of this thought pattern was a closed end system. Being closed ended there was no means of easily extending the system. This is one of the main hindrances to making systems interoperable.

### *c. Standard Data Format*

The DOD legacy systems were designed at a time when there was little or no standardization of data. This barrier alone makes it hard to communicate between the legacy systems. This issue was known, but little has been accomplished in this area. Until now the need for exchange of information has not been a big issue. Like many issues, if there is no motivation, there is no effort to solve the issue.

In order for the standardization issue to be addressed, there has to be interest from the decision-makers. This is the case currently. Now there is a need for communication in area of data that wasn't the case in the past when the DOD legacy systems were developed.

Just to standardize data is not enough. Efforts have to be made to ensure that common data has a well-understood common meaning among the users and developers. This is very important because the connotation of the same information may differ within DOD depending on what department is using the information.

### 3. Potential Solutions To DOD Interoperability

In order to cross the hurdles of the barriers discussed above, a bridge has to be created by which the data of the legacy systems may cross over. [REN] states this can be done by performing three tasks: develop a model and data elements for the shared data, convert the legacy data values to the new format, and modify the application programs to use the shared data server and its schema.

Development of a model to allow the data to be shared requires the development of a process that would allow data sharing. In order to share anything one must understand what is being shared. This would require standard data elements and interpretations as discussed earlier. Standardization is of utmost importance when it comes to data sharing.

Secondly, the old data has to be modified to fit into the derived standard form. The conversion of the old data into new data may require different translation tools based on the DBMS of the system. Nevertheless it has to be done.

Lastly, the application software needs to be modified to use the new shared data. This would require the developers to take a closer look at the data elements to ensure that necessary changes have been made.

DOD is taking steps to provide a means of making data more interoperable via XML. The Defense Information Systems Agency (DISA) is guiding the Defense

Information Infrastructure Common Operating Environment (DII COE) to establish an XML management framework and web based registry for DOD XML users. This Framework is designed to address the causes of non-interoperable XML and ballooning management overhead from proliferation of XML groups. One vehicle to address data disambiguation (collisions/conflict) is the namespace concept. The DII COE XML Registry defines a namespace as a collection of items that share an interest in a particular problem domain or practical application. This registry allows namespaces to be published while being provided to outsiders for use if necessary. At the same time it allows persons to add namespaces to the registry. In order to keep consistency within namespaces, a DII COE namespace has certain criteria to meet in order to be added to the registry.

A DID COE Namespace must:

- Have a sponsor
- Have a Point of Contact
- Be willing to make its communal defintions visible
- Be willing to finance/staff namespace adminstration
- Be chartered by the COE Chief Engineer (with CRCB oversight)
- Involve (at some point) specific COE system implementations. [DII COE1]

## B. IS XML THE SOLUTION?

### 1. What is XML?

XML is a markup language for documents containing structured information. XML is a subset of SGML. Its goal is to enable generic SGML to be served, received, and processed on the web in the way that is now possible with HTML. XML has been

designed for ease of implementation and for interoperability with both SGML and HTML.

## 2. Why XML?

Why XML? Because XML provides a means by which structured data can be transmitted and stored. XML would allow the developer to query data sources and not be controlled by the different DBMS from which the data is coming. Once the developer knows the schema of the XML document, it can retrieve the information and then display it in the user-preferred format.[MASXML]

XML also is not affected by changes to the data sources, provided the changes are not to a piece of information the XML document needs.

## 3. XML Advantages

One of the main advantages with XML is implied in its name. It is extensible. With XML the developer is allowed to create based on the need of the user.

Unlike relational databases, XML allows for the ease of changes to the data structure.

## 4. XML Pitfalls

XML falls short of DBMS offered capability in the area of querying, processing, and viewing of the data. Unlike DBMS, XML does not have the capability of data manipulation that is available with DBMS.

Efficiency is not a top priority with XML. XML focuses in on simplicity. Therefore, XML and DBMS have their own niche. However, some of the issues are being addressed by other facilities associated with XML, such as the style sheet language, XSL, and the translator, XST.

# III. DBMS OF INTEREST

## A. INTRODUCTION OF THE DATABASES

The DOD legacy systems of interest vary between four major DBMSs. The DBMSs and databases of interest are as follows:

| Database | DBMS |
|---|---|
| Army Advanced Field Artillery Tactical Data System (AFATDS) | Interbase |
| Global Command & Control System (GCCS) Integrated Intelligence And Imagery (I3) | Sybase |
| GCCS Track Database Manager (TDBM) | Oracle |
| Joint Common Data Base (JCDB) | Informix |

## B. AFATDS (INTERBASE)

AFATDS is a network of computer workstations that process and exchange information from the forward observer to the fire support element for all fire support assets (i.e. field artillery, attack helicopters, naval gunfire, mortars, and close air support). AFATDS has some of the following features: automatic processing of fire requests, generation of multiple tactical fire solutions for missions, monitoring of mission execution, and support for the creation and distribution of fire plans. It is a multi-service (US Army and USMC), joint and combined forces fire support C3 system.

In the past, fire support was conducted manually or semi-automatically. This took more time, was less accurate and sometimes caused loss of life. AFATDS is a fully

automated fire support system that minimizes the sensor-to-shooter timeline and increases the hit ratio. Together, these two benefits ensure mission success and save lives.

AFATDS uses Interbase as its DBMS and was not analyzed by the author of this thesis. This database was not analyzed because the necessary information could not be obtained in a timely manner.

## C. GCCS I3 (SYBASE)

The GCCS Integrated Intelligence and Imagery (I3) system is a tool that overlays Defense Intelligence Agency data, Order of Battle, and targets on imagery using the Joint Mapping Tool Kit (JMTK). I3 enhances GCCS with the ability to access military intelligence imagery assets. It provides necessary intelligence features to the warfighter and consists of 44 segments which comprise several key databases and activities.

GCCS I3 uses Sybase as its DBMS and was investigated by another member of the research team. His findings are published in [HIN99].

## D. GCCS TDBM (ORACLE)

The GCCS Track Database Manager (TDBM) has a mission application that manages track information. This track information is displayed on a Windows 95/98/NT environment via software called Command and Control Personal Computer (C2PC).

C2PC features include:

- Provides support for a wide variety of digital map data.
- Provides a tactical unit/target database and editor.
- Provides support for mapping projections.
- Supports over 200 mapping datums.

- Provides full track database add, edit, and delete capabilities.

GCCS Track Database Management (TDBM) System was developed with Oracle and there was not enough information to research its potential to use XML as a means of interoperability.

## E. JCDB (INFORMIX)

The Joint Common Database (JCDB) is the database that this thesis focuses on. The JCDB uses informix as its DBMS. Therefore, this thesis will attempt to show that XML can be used as a means of communicating with the other DBMS mentioned above. The JCDB is made up of data that both the Army and Joint C2 systems use. Therefore, it is a viable database to consider when discussing data interoperability. The JCDB contains information in the following areas: Logistics, Maneuvering, Air Defense, Intelligence, Network Management, Fire Support, and GCCS-Army.

The JCDB is made up of the following suite of products:

1. The Joint Data Model (JDM) is used to capture system, user and interoperability requirements.
2. The Joint Data Dictionary (JDD) is a dictionary of all the data elements (attributes) utilized in the JCDB.
3. The Subscribe and Receive Data Distribution Mechanism (SRDDM) is a data distribution software used to distribute changes to JCDB data to all other active JCDB's in a LAN configuration.
4. The JCDB Application Programming Interfaces (APIs) are designed to simplify access to the JCDB by user systems.

Though the JCDB has considered interoperability, it is a more narrow view than the one that is needed. More consideration is needed in the area of interoperability between JCDB and other DOD legacy systems. This thesis will investigate this point using XML as a potential solution.

# IV. ANALYSIS OF XML TO SOLVE INFORMIX DATA INTEROPERABILITY

## A. ANALYSIS OF XML LITERATURE

After reviewing literature on XML via many sources (i.e. web, books, magazines, etc.) they all point to the same thing, XML can solve the interoperability problem of exchanging information between the DOD legacy systems. XML is considered a meta-language. That means it represents data that describes other information, such as how it is structured and how it can be exchanged between servers and clients via a network. By providing these features, XML makes it much simpler for programmers to develop applications that move information from system to system.

Whenever an application receives an XML file, it gets descriptions about how the data is structured. That way, a program can more easily determine how to process the data.

As mentioned earlier, the focus of this thesis is the JCDB database, which was developed using the Informix DBMS. Therefore, the focus will be on how XML can be used along with Informix to transfer data between other DBMSs.

## B. XML A FEASIBLE SOLUTION?

In a quest to find out if XML is a solution for communication between DOD legacy systems, specifically Informix, it was found that one of the best tools to use would be Informix Internet Foundation.2000. Due to the availability of the proper equipment to test this tool, this thesis will only describe the tool and its parts.

The Informix Internet Foundation.2000 tool is made up of 5 sub-products. The Informix Internet Foundation.2000 products are explained next.

### 1. Informix Internet Foundation.2000

Based on the trend going toward XML, Informix has found a need to follow that trend. In doing so, Informix Software has developed a product to assist in the area of XML. Informix's XML strategy is to make Informix fast and easy to build high performance XML data-driven Internet applications that adhere to and promote open standards. [INFSITE] As discussed in an earlier section, open standards would help eliminate one of the barriers that hinder the legacy systems from being interoperable. The Informix Internet Foundation.2000 product family will exploit XML to provide smart distributed data for applications.

The tool that allows Informix to exploit XML is Informix Internet Foundation.2000. Informix Foundation.2000 has a suite of tools designed to help users to expand their applications to the Web quickly and reliably.

Informix Internet Foundation.2000 is designed for development and deployment of component based applications. It consists of a core database engine, Informix Dynamic Server.2000, and a set of tools to help facilitate internet application development and deployment.

### 2. Informix Dynamic Server.2000

Dynamic Server.2000 is a multi-threaded, object-relational data engine designed to deliver breakthrough database scalability, manageability, and performance. Informix Dynamic Server.2000 incorporates extensibility directly into the database. Informix Dynamic Server.2000 benefits the users with an advantage of object-oriented technology

and unlimited extensibility. Informix Dynamic Server.2000 can extend even further with DataBlade Module technology.

Dynamic Server.2000 is a database server that enables you to create and use existing DataBlade module packages to encapsulate specialized data types, operations that process the data, and access methods that index the data. DataBlade modules can be plugged into this database server to work with different types of Informix data. [INFSITE]

### 3. Informix Web DataBlade Module

Applications that access data from one or more sources may want to receive the data in an XML format, so that applications are insulated from dealing with multiple systems that involve different formats. The Web DataBlade module of Internet Foundation.2000 enables users to generate dynamic XML data and documents using a familiar SQL interface. With Web DataBlade users can easily and seamlessly publish XML data over the Internet. Thanks to Web DataBlade XML can be delivered to a Web browser or an enabled application that utilizes the HTTP protocol.

The Web DataBlade module provides a complete set of SQL functions, data types, tags, and client applications that enables the user to create Web applications that incorporate data retrieved from Informix databases.

When the Web DataBlade module is used, you create HTML pages that Web DataBlade module specific tags (also called AppPage Tags) and functions that execute your SQL queries dynamically and format the results for the Web. The pages created are

referred to as Application Pages (AppPages). The Web DataBlade module also enables users to generate XML data and elements over the Internet.

The AppPages discussed earlier are stored in the Informix database. The web application that uses the Web DataBlade module, first retrieves the AppPage from the database, then passes the AppPage through a SQL function that interprets the special AppPage tags and functions, usually to retrieve or update data from the database tables and formats the results.

Figure 1 below depicts the Web DataBlade process. As with any web application in order to invoke an application you type in an URL or click on a link. Once one of these actions have been taken in a Web DataBlade module application, the Web browser invokes a URL to request information stored in the Informix database. This request is made to the Web server which in turns invokes a WebDriver, which establishes a database connection requesting the information from the database. At this point the WebExplode function takes control to retrieve the requested information dynamically and returns it to the WebDriver as HTML. Finally, WebDriver returns the new information to the Web server, which in turn returns the HTML page to be displayed by the Web browser.

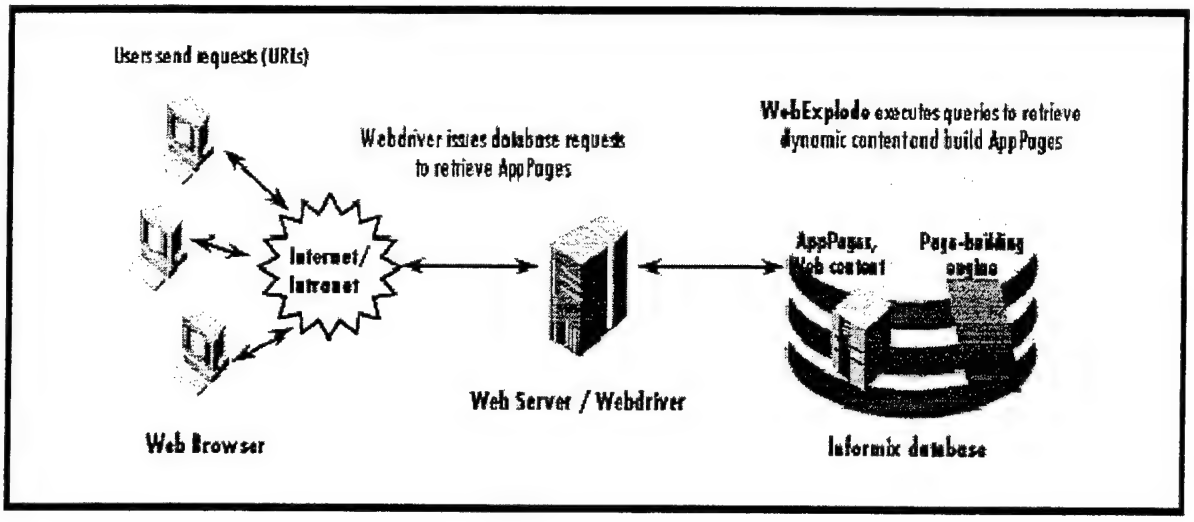



Figure 1 The Web DataBlade module retrieval process.

Though Web DataBlade is a tool that can be used to convert Informix data to XML data, there are certain system requirements that have to be followed. These requirements are for Web DataBlade module version 4.0.

The following operating systems are supported:

- Sun SPARC running Solaris 2.5.1 of higher.

The following database servers are supported:

- Informix Dynamic Server with Universal Data Option, version 9.14.UC4 or higher
- Informix Internet Foundation .2000, version 9.20.UC1 or higher
- Informix Dynamic Server.2000, version 9.20.UC1 or higher.

The Informix Client Software Development Kit,, version 2.10.UC1 or higher, is required for the WebDriver client.

The Web DataBlade module is compatible with any Web browser that conforms to the HTML 3.0 specification for tables and forms.

### 4. Informix Hierchical XML Data Storage

The Informix Hierchical XML Data Storage feature of Internet Foundation.2000 allows users to be able to import, export, store, query, and index XML structures in their own hierarchical format. One advantage is that it has better performance over other approaches to storing XML documents in flat relational tables. This product can be used by any Java XML parser and integrates XML with the pre-existing database schema.

Hierchical XML Data Storage stores, indexes, and queries XML documents in accordance with a pre-specified or default XML-to-Object-oriented Relational Database Management System (ORDBMS) mapping. The XML document is usually stored in tables but can also be stored as Binary Large Objects (BLOBs).

### 5. Informix XML Metadata Repository

Informix is pursuing the concept of an XML Metadata Repository. It will create a standard XML representation of database schema and other kinds of metadata. The concept is looking at providing an XML metadata repository through the Informix Dynamic Server.2000 database server.

## C. PHP4 ANOTHER POSSIBLE SOLUTION

During the research another possible solution for data interoperability was found. This solution involves using PHP Hypertext Preprocessor (PHP) as a solution.

Technically, PHP4 is a cross-platform, HTML-embedded, server-side web scripting language[PHPNET]. In more detail:

- Cross-Platform means one can run PHP4 code, without alteration, on computers running many different operating systems. For example, a PHP4 script that runs on Linux will generally run on Windows as well.
- PHP4 scripting language can be embedded directly into the HTML code.
- PHP4 code runs on a server – specifically, a web server.
- PHP4 programs run via a web browser. The web server on which the PHP4 code reside runs the program, sending any resulting output to the browser. [PHP400]

PHP4 scripting does offer an interesting benefit as well. PHP4 will allow interoperability with the DOD legacy databases in their existing formats. With this scripting language you can retrieve data from the database as long as the correct field names are known.

On the other hand, a big disadvantage is evident when there is a change made to the database schema and the PHP4 code is not changed to reflect the change. Therefore, configuration management is of the utmost importance.

As stated earlier, since this thesis main focus is on the JCDB database, some data elements were used from the JCDB database schema to give an example of how PHP4 works. Though JCDB uses Informix as its DBMS, this example uses Microsoft Access as the DBMS. This is due to the fact that an Informix DBMS was not accessible by the author of this thesis. Therefore, the example will show how it can be done using Microsoft Access. Attention will be paid to how Informix as well as the other DBMSs that the legacy DOD databases use can interact with PHP4 also.

### 1. PHP4 with Microsoft Access

As stated earlier, in order for PHP4 to work it has to be used with a web server. Access DBMS has a Web Server that comes with Microsoft Windows called Personal Web Server (PWS). By installing the PWS, PHP4 can be used alongside the PWS. PHP4 scripting language can be downloaded, free of charge, from http://php4win.de/ website.

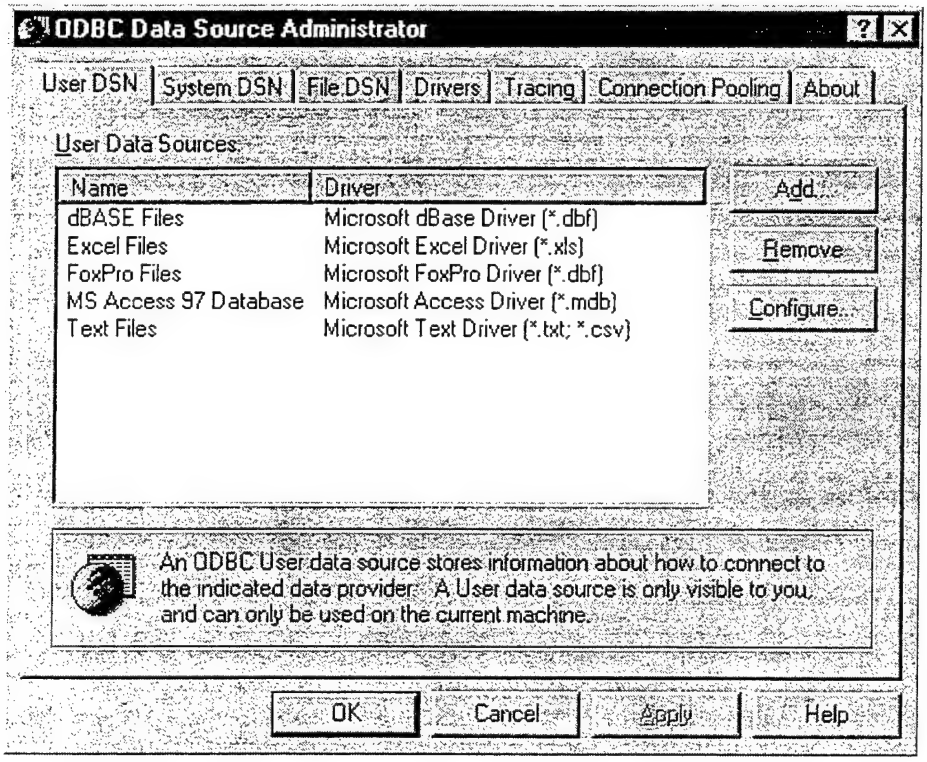
It is recommended that once both PWS and PHP4 have been installed on the client machine, to test both to ensure they are working together properly.

Once working properly, an example using Microsoft Access was created. The main objective of this example is to show that information can be accessed via HTML and PHP4. The PHP4 scripting language has different functions based on the type of DBMS that is being used. In this example, the Open Database Connectivity (ODBC) functions are used to interact with Microsoft Access. In order for this to work properly, it was necessary to setup a Data Source Name (DSN) within the Windows 98 environment.

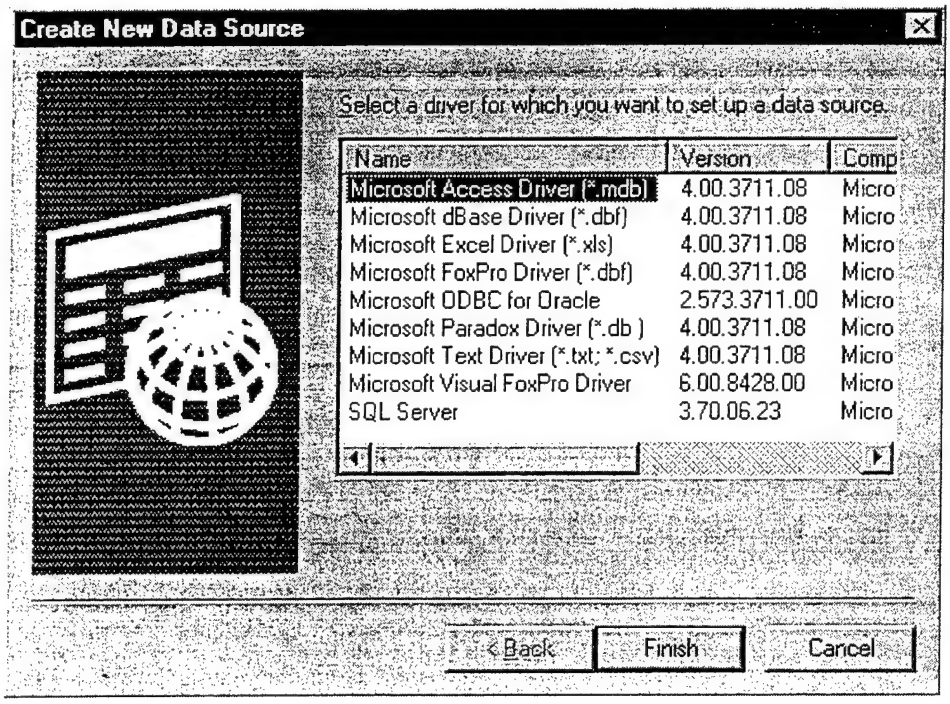
ODBC is not a full database abstraction layer, that's to say it manages the mechanics of connecting to databases, but doesn't provide a translation service. "Think of the ODBC data source manager as a telephone switchboard: it can put you through to the Data Source Name (DSN) you requested, but once you're connected, it doesn't get involved in helping you talk to the server. This means you'll still need to know the specifics of the SQL syntax used by the server you're connecting to, and that's going to be different from one server to the next." [PHP400]

The DSN is very important to the connectivity to the DBMS of interest because this is one of the parameters in the connectivity functions of PHP4. Here is an example of the steps it takes to create a DSN in the Windows environment. It required that an ODBC

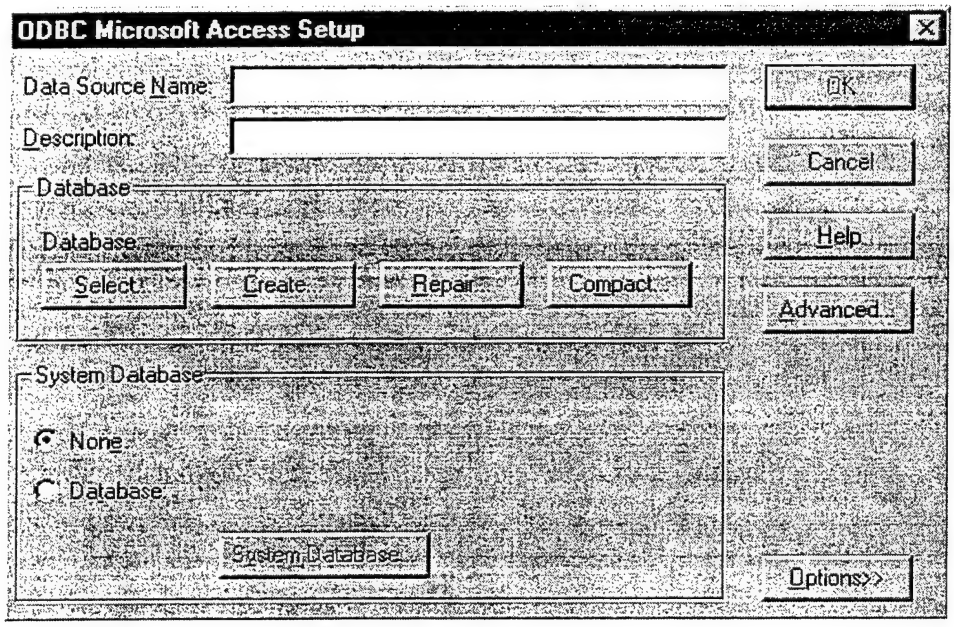
data source manager be identified. From the control panel of Windows 98 the ODBC Data Sources was selected:




At this point, a data source under the System DSN tab has to be created. This will make it available to all programs running on the computer, including PHP. Once the add button is clicked, a screen with the available data sources will appear:




Now the database driver for the appropriate DBMS can be selected. Once the click button is clicked, a screen to give a DSN will appear:



The DSN will be set once OK is selected. Now the DBMS will have connectivity through PHP4.

One the DSN is setup your computer is ready to run PHP4. The PHP4 script can be created using any text editor (i.e. Notepad, Wordpad). Using a text editor the following PHP4 script was written to test the connectivity as well as the retrieval capabilities of PHP4 to Microsoft Access:

```
<?php
// connect to a DSN "thesis" (user and password could be entered if available)
$connect = odbc_connect("thesis", "", "");

// query the target_info table in the JCDB example database
$query = "SELECT trgt_airdef_cd, air_engagement, trgt_dispo_cd, trgt_engage_assess,
munition_assessment, trgt_list_typ_cd, target_engagement, trgt_org_assc_cd,
trgt_eng_org_assc,
trgt_source_cd, supported_target, trgt_symbol_cd, cndt_trgt FROM target_info";

// perform the query
$result = odbc_exec($connect, $query);

// fetch the data from the database
while(odbc_fetch_row($result)){

  $trgt_airdef_cd = odbc_result($result, 1);
  $air_engagement = odbc_result($result, 2);
  $trgt_dispo_cd = odbc_result($result, 3);
  $trgt_engage_assess = odbc_result($result, 4);
  $munition_assessment = odbc_result($result, 5);

//print the 5 fields from the target_info table (output goes to the web browser being used)
  print("$trgt_airdef_cd  $air_engagement $trgt_dispo_cd $trgt_engage_assess
$munition_assessment \n");
}

// close the connection
odbc_close($connect);
?>
```

This PHP4 script was included in a HTML file to run via the web browser. The complete listing of the HTML file is found in Appendix A. This PHP4 script file allows the web browser to retrieve information from the MS Access file.

The MS Access files include dummy target information. The fields included in the example target info table of the xml_example database are listed in detail in Appendix B. This is only an example of what is possible using PHP4. MS Access was only used as an example to show that it is possible due to the lack of equipment and software to prove it in Informix, the JCDB DBMS.

The following paragraphs will give a hint as to what has to be changed in the PHP4 script in order for this script to interact with other DBMS of the DOD legacy systems (i.e. Informix, Oracle, Sybase, and Interbase). The key functions to make the PHP4 script interact with the DBMS of interest is found in Appendix C. The details of these functions can be found at http://www.php.net/distributions/bigmanual.html.

### 2. PHP4 with Informix, Oracle, Sybase, and Interbase

In order to interact with the DOD legacy systems DBMSs there are four main functions that have to be addressed as is shown in the MS Access example. They are:

a. Connect with the DBMS

b. Perform the query

c. Retrieve the selected data

d. Make data available to program

e. Close the connection.

Appendix C has the necessary functions for each of the DBMSs of interest. It should be just a matter of replacing the ODBC functions, in the MS Access example, with the necessary functions for the DBMS of interest.

### 3. PHP4 with XML

Though the focus of this thesis is mainly on XML capabilities to allow interoperability between DOD legacy databases, PHP4 and HTML offers an alternative to the XML approach. However, PHP4 does have XML capabilities. There are many ways of accessing XML data, and it's all dependent on the parsing model. One in particular is the Simple API for XML (SAX) model. SAX is considered to be an event-driven parsing model. With an event-driven parsing model, the parser reads through the document and sends signals out when certain events occur. In other words, when the parser gets to the beginning of a new element (an event) it sends out a signal, along with some data, that is pertaining to the event in question.

The advantage to this type of model is that the entire XML document is not read before the data can be used. PHP4 has an XML parsing model called PHP4 API for XML. This parsing model is very similar to SAX.

PHP4 API for XML can be used to parse the XML data. Therefore some other action has to be taken to convert the legacy data into an XML format. One suggestion would be the use of Informix Foundation.2000 to perform this task. As stated in the discussion of the Web DataBlade Module, it can be used to convert the Informix data into XML data.

This thesis presents an example of how PHP4 can be used in conjunction with XML to parse an XML document to be presented using a web browser. The XML data discussed in the previous statement needs to be in an XML format prior to use by PHP4.

The first step requires a well-formed XML document, well-formed in the since that the document takes the following rules into consideration.

- Tags must be nested properly.
- All start tags must have end tags.
- You must use quotation marks correctly for tag attributes.
- You must be careful using certain characters ('&' and '<' specifically) in your data.

The data being used in this example is target type data that was taken from a previous thesis. [HINA] A list of the XML document is found in Appendix E.

The XML Document lacks some important information needed about the specifics of the data. When looking at a particular document, there is a need to ascertain which tags are required, what kinds of data they should hold, what order they should come in, and which tags are not allowed. This is done with a Document Type Definition or DTD. Appendix F gives a view of the DTD that is used in this example.

Now that an error-free DTD exists, we can now take a look at the PHP4 API script that can be used to present the XML data via a web browser. As mentioned earlier, PHP4 API for XML is similar to SAX. Both are event driven and have very similar functions for assigning callbacks to the events produced by the parser.[SAX]

There are basically five functions from the PHP4 API that are used to parse the example file. They are:

| **Function**: | **Description**: |
|---|---|
| int xml_parser_create() | Creates an XML parser reference by int. This will return false if the parser is not created. |
| int xml_set_element_handler(int parser, string startElementHandler, string endElementHandler) | Sets up callback functins for the "begin element" and "end element" events. The functions are referenced by the strings startElementHandler and endElementHandler. Returns *true* if the functions are set successfully, *false* otherwise. |
| Int xml_set_character_data_handler(int parser, string characterDataHandler) | Sets up the callback function to handle character data between elements. This function is referenced by the string characterDataHandler. Returns True if the function is set successfully, False otherwise |
| int xml_parse(int parser, string data[,int isFinal]) | Starts parsing the information represented by data. The isFinal argument is an optional argument that may be used to tell the parser when to quit. Returns True as long as data is parsed successfully. |
| string xml_parser_free (int parser) | Frees memory used by parser. Returns True if the memory is freed successfully, False otherwise. |

Before continuing with the example, the three callback functions in the table above need to be discussed. The function *startElementHandler* must have the following format: *startElementHandler*( int *parser*, string *name*, array attribs). Here, *parser* is the pointer to the XML parser that came from xml_parser_create() function. The string *name* is the name of the element that is being parsed.

The *endElementHandler* function has two arguments: *parser* and *name*. *Parser* is the pointer to the XML parser. *Name* is the name of the element that was just examined.

Finally, the *characterDataHandler* function looks like: *characterDataHandler*(int *parser*, string *data*). The *parser* is the familiar XML parser pointer as before. The *data* string is all the character data within the current element.

Now that all of the event-driven functions have been covered the PHP4 scripting file can be explained. First the XML file must be opened by using an if statement and fopen command:

if( ! ($fp = fopen( "*filename*", R ))) .

Since there are multiple targets in the example, there is a need to set global variables that can be used to keep track of the parser as it goes through the application:

```
$target_counter = 0;
$target_data = array();
```

There is also a need to be able to tell the character data handler what to do. Therefore the current tag state has to be tracked as well. This is done by initializing a variable: $xml_current_tag_state = ‘’; .

Now the 3 callback functions discussed earlier have to be defined. One important note is that all of the callback functions must be defined before they are applied to the XML parser.

This example requires the startElementHandler function to make the global variables available and set the xml_current_tag_state equal to the element name. The function is as follows:

```
function startElementHandler( $parser, $element_name, $element_attribs )
{
        global $target_counter;
        global $target_data;
        global $xml_current_tag_state;

        $xml_current_tag_state = $element_name;
}
```

The endElementHandler function makes the global variables available, sets the xml_current_tag_state to blank, and increases the target counter by 1. This counter will keep track of the target(s). The endElementHandler function is as follows:

```
function endElementHandler( $parser, $element_name )
{
        global $target_counter;
        global $target_data;
        global $xml_current_tag_state;

        $xml_current_tag_state = '';


        if( $element_name == "TARGET" )
        {
                $target_counter++;
        }
}
```

This endElementHandler Function will execute when the parser gets to the end of an element.

The final function is the character data handler to perform action on the data collected thus far. The characterDataHandler function will set the global variables for use and check to see if an element name exists. If an element name exists it puts the data into the target_data global associative array for use later, otherwise it returns.

All the legwork has been done. The only thing left to do is to parse the data with an XML parser. The code necessary to perform this parsing is found in Appendix G. Once the data is parsed, the only thing remaining is to write HTML code to display the output in a web browser.

In summary of this chapter, I found that PHP4 API for XML has a capability to display XML data on a web page. In order to accomplish this, I found that PHP4 requires the XML data to already be in an XML format. It is suggested that some other method be used to convert the data from the legacy systems to XML format. When this is accomplished, it only becomes a matter of making minor changes to the PHP4 scripting program to make it work for other documents.

# V. CONCLUSION AND FUTURE RESEARCH POSSIBILITIES

In this thesis two major areas were investigated to answer the question of interest. First of which, "Can XML help with the data interoperability problem that exists with the DOD legacy systems?" If so, how can it done. A second question, "Is there another possible solution to allow data interoperability between the DOD legacy system?" is also addressed.

Based on the research performed for this thesis, the first question mentioned in the paragraph above has a prospective answer. There are many tools that are available to assist in this area. This thesis only focuses in on one of them. There have been other theses that have investigated other XML tools. [HINA00] This thesis only focused in on XML tools that may help with Informix. Informix was chosen because it was the DBMS that was used to develop JCDB. The one tool that was investigated was Internet Informix Foundation.2000. After investigating Internet Informix Foundation.2000, it was found that it was made up of multiple products that can make it easy to use XML to make Informix data available to other sources. For further research, it is suggested that someone with the right software and equipment to actually test the software for its capabilities to meet the need of data interoperability.

The second question concerning other possibilities was investigated as well. It was found that there is another possibility to be used with HTML that will provide a more immediate solution. More immediate, because it will allow for interaction with the current DBMS.

This thesis includes two examples of how this can be done. Though the first example uses Microsoft Access, it can be used with other DBMSs as well. The one

disadvantage to this approach is that it is schema dependent. A change in the schema could effect the output of the PHP4 script. Since the script uses a SQL statement, it is totally dependent on the fields and their definition. If there is a change in one of the fields that are used it the script, the PHP4 script has to be changed to reflect that change. This solution's configuration has to be controlled a little more carefully. But as I mentioned earlier, this is a short-term fix if you are looking for something that can be developed more quickly.

The second example demonstrates that PHP4 in conjunction with XML can be used to make data available. As stated in this example the input for PHP4 is a well-formed XML document. Though this thesis used an XML document from previous research, there is still a research opportunity to discover a means of converting the legacy system data into XML data.

## APPENDIX A (HTML CODE)

```
<HTML>
<HEADER></HEADER>
<BODY>

This is a test to see if I can connect to the access database.

<?php
// connect to a DSN "mydb" with a user and password "marin"
$connect = odbc_connect("thesis", "", "");

// query the user's table for name and surname
$query = "SELECT trgt_airdef_cd, air_engagement, trgt_dispo_cd, trgt_engage_assess,
munition_assessment, trgt_list_typ_cd, target_engagement, trgt_org_assc_cd,
trgt_eng_org_assc,
trgt_source_cd, supported_target, trgt_symbol_cd, cndt_trgt FROM target_info";

// perform the query
$result = odbc_exec($connect, $query);

// fetch the data from the database
while(odbc_fetch_row($result)){

 $trgt_airdef_cd = odbc_result($result, 1);
 $air_engagement = odbc_result($result, 2);
 $trgt_dispo_cd = odbc_result($result, 3);
 $trgt_engage_assess = odbc_result($result, 4);
 $munition_assessment = odbc_result($result, 5);

 print("$trgt_airdef_cd  $air_engagement $trgt_dispo_cd $trgt_engage_assess
$munition_assessment \n");
}

// close the connection
odbc_close($connect);
?>

</BODY>
</HTML>
```

THIS PAGE INTENTIONALLY LEFT BLANK

# APPENDIX B (MICROSOFT ACCESS TABLE DEFINITION)

| Field Name | Data Type | Description |
|---|---|---|
| trgt_airdef_cd | Number | The code that denotes the ait defense weapons type at a TARGET for the specific AIR-ENGAGEMENT. |
| air_engagement | Number | The code that denoteds the type of Air Target Engagement. |
| trgt_dispo_cd | Number | The code which denoted the state of a TARGET after it has been ENGAGED. |
| trgt_engage_assess | Number | |
| munition_assessment | Number | |
| trgt_list_typ_cd | Number | |
| target_engagement | Number | |
| trgt_org_assc_cd | Number | The code that denotes the manner in which a target engageent is associated with a specific organizati |
| trgt_eng_org_assc | Number | |
| trgt_source_cd | Number | The code which denotes the source class of a SUPPORTED-TARGETs identity. |
| supported_target | Number | |
| trgt_symbol_cd | Number | The GSD code which is provided for reference to TARGET symbols. |
| cndt_trgt | Number | |
| | | |
| | | |
| | | |
| | | |
| | | |

**THIS PAGE INTENTIONALLY LEFT BLANK**

## APPENDIX C (PHP KEY FUNCTIONS)

### ODBC Functions

odbc_autocommit — Toggle autocommit behaviour

odbc_binmode — Handling of binary column data

odbc_close — Close an ODBC connection

odbc_close_all — Close all ODBC connections

odbc_commit — Commit an ODBC transaction

odbc_connect — Connect to a datasource

odbc_cursor — Get cursorname

odbc_do — Synonym for odbc_exec()

odbc_exec — Prepare and execute a SQL statement

odbc_execute — Execute a prepared statement

odbc_fetch_into — Fetch one result row into array

odbc_fetch_row — Fetch a row

odbc_field_name — Get the columnname

odbc_field_num — Return column number

odbc_field_type — Datatype of a field

odbc_field_len — Get the length (precision) of a field

odbc_field_precision — Synonym for odbc_field_len()

odbc_field_scale — Get the scale of a field

odbc_free_result — Free resources associated with a result

odbc_longreadlen — Handling of LONG columns

odbc_num_fields — Number of columns in a result

odbc_pconnect — Open a persistent database connection

odbc_prepare — Prepares a statement for execution

odbc_num_rows — Number of rows in a result

odbc_result — Get result data

odbc_result_all — Print result as HTML table

odbc_rollback — Rollback a transaction

odbc_setoption — Adjust ODBC settings. Returns false if an error occurs, otherwise true.

odbc_tables — Get the list of table names stored in a specific data source. Returns a result identifier containing the information.

odbc_tableprivileges — Lists tables and the privileges associated with each table

odbc_columns — Lists the column names in specified tables. Returns a result identifier containing the information.

odbc_columnprivileges — Returns a result identifier that can be used to fetch a list of columns and associated privileges

odbc_gettypeinfo — Returns a result identifier containing information about data types supported by the data source.

odbc_primarykeys — Returns a result identifier that can be used to fetch the column names that comprise the primary key for a table

odbc_foreignkeys — Returns a list of foreign keys in the specified table or a list of foreign keys in other tables that refer to the primary key in the specified table

odbc_procedures — Get the list of procedures stored in a specific data source. Returns a result identifier containing the information.

odbc_procedurecolumns — Retrieve information about parameters to procedures

odbc_specialcolumns — Returns either the optimal set of columns that uniquely identifies a row in the table or columns that are automatically updated when any value in the row is updated by a transaction

odbc_statistics — Retrieve statistics about a table

### Oracle Functions

Ora_Bind — bind a PHP variable to an Oracle parameter

Ora_Close — close an Oracle cursor

Ora_ColumnName — get name of Oracle result column

Ora_ColumnSize — get size of Oracle result column

Ora_ColumnType — get type of Oracle result column

Ora_Commit — commit an Oracle transaction

Ora_CommitOff — disable automatic commit

Ora_CommitOn — enable automatic commit

Ora_Do — Parse, Exec, Fetch

Ora_Error — get Oracle error message

Ora_ErrorCode — get Oracle error code

Ora_Exec — execute parsed statement on an Oracle cursor

Ora_Fetch — fetch a row of data from a cursor

Ora_Fetch_Into — Fetch a row into the specified result array

Ora_GetColumn — get data from a fetched column

Ora_Logoff — close an Oracle connection

Ora_Logon — open an Oracle connection

Ora_pLogon — Open a persistent Oracle connection

Ora_Numcols — Returns the number of columns

Ora_Numrows — Returns the number of rows

Ora_Open — open an Oracle cursor

Ora_Parse — parse an SQL statement

Ora_Rollback — roll back transaction

## Sybase Functions

sybase_affected_rows — get number of affected rows in last query

sybase_close — close Sybase connection

sybase_connect — open Sybase server connection

sybase_data_seek — move internal row pointer

sybase_fetch_array — fetch row as array

sybase_fetch_field — get field information

sybase_fetch_object — fetch row as object

sybase_fetch_row — get row as enumerated array

sybase_field_seek — set field offset

sybase_free_result — free result memory

sybase_get_last_message — Returns the last message from the server

sybase_min_client_severity — Sets minimum client severity

sybase_min_error_severity — Sets minimum error severity

sybase_min_message_severity — Sets minimum message severity

sybase_min_server_severity — Sets minimum server severity

sybase_num_fields — get number of fields in result

sybase_num_rows — get number of rows in result

sybase_pconnect — open persistent Sybase connection

sybase_query — send Sybase query

sybase_result — get result data

sybase_select_db — select Sybase database

## Interbase Functions

ibase_connect — Open a connection to an InterBase database

ibase_pconnect — Creates an persistent connection to an InterBase database

ibase_close — Close a connection to an InterBase database

ibase_query — Execute a query on an InterBase database

ibase_fetch_row — Fetch a row from an InterBase database

ibase_fetch_object — Get an object from a InterBase database

ibase_field_info — Get information about a field

ibase_free_result — Free a result set

ibase_prepare — Prepare a query for later binding of parameter placeholders and execution

ibase_execute — Execute a previously prepared query

ibase_trans — Begin a transaction

ibase_commit — Commit a transaction

ibase_rollback — Rolls back a transaction

ibase_free_query — Free memory allocated by a prepared query

ibase_timefmt — Sets the format of timestamp, date and time type columns returned from queries

ibase_num_fields — Get the number of fields in a result set

ibase_errmsg — Returns error messages

**Informix Functions**

ifx_connect — Open Informix server connection

ifx_pconnect — Open persistent Informix connection

ifx_close — Close Informix connection

ifx_query — Send Informix query

ifx_prepare — Prepare an SQL-statement for execution

ifx_do — Execute a previously prepared SQL-statement

ifx_error — Returns error code of last Informix call

ifx_errormsg — Returns error message of last Informix call

ifx_affected_rows — Get number of rows affected by a query

ifx_getsqlca — Get the contents of sqlca.sqlerrd[0..5] after a query

ifx_fetch_row — Get row as enumerated array

ifx_htmltbl_result — Formats all rows of a query into a HTML table

ifx_fieldtypes — List of Informix SQL fields

ifx_fieldproperties — List of SQL fieldproperties

ifx_num_fields — Returns the number of columns in the query

ifx_num_rows — Count the rows already fetched a query

ifx_free_result — Releases resources for the query

ifx_create_char — Creates an char object

ifx_free_char — Deletes the char object

ifx_update_char — Updates the content of the char object

ifx_get_char — Return the content of the char object

ifx_create_blob — Creates an blob object

ifx_copy_blob — Duplicates the given blob object

ifx_free_blob — Deletes the blob object

ifx_get_blob — Return the content of a blob object

ifx_update_blob — Updates the content of the blob object

ifx_blobinfile_mode — Set the default blob mode for all select queries

ifx_textasvarchar — Set the default text mode

ifx_byteasvarchar — Set the default byte mode

ifx_nullformat — Sets the default return value on a fetch row

ifxus_create_slob — Creates an slob object and opens it

ifxus_free_slob — Deletes the slob object

ifxus_close_slob — Deletes the slob object

ifxus_open_slob — Opens an slob object

ifxus_tell_slob — Returns the current file or seek position

ifxus_seek_slob — Sets the current file or seek position

ifxus_read_slob — Reads nbytes of the slob object

ifxus_write_slob — Writes a string into the slob object

THIS PAGE INTENTIONALLY LEFT BLANK

## APPENDIX D (ACRONYMS)

| | |
|---|---|
| Army Advanced Field Artillery Tactical Data System | AFATDS |
| Application Programming Interface | API |
| Binary Large Objects | BLOB |
| Commercial Off the Shelf | COTS |
| Command and Control | C2 |
| Database Management System | DBMS |
| Department of Defense | DOD |
| Data Source Name | DSN |
| Extensible Markup Language | XML |
| Global Command and Control | GCCS |
| HyperText Markup Language | HTML |
| Joint Battlefield Infosphere | JBI |
| Joint Common Database | JCDB |
| Joint Data Dictionary | JDD |
| Joint Data Model | JDM |
| Microsoft | MS |
| Open Database Connectivity | ODBC |
| PHP Hypertext Preprocess | PHP |
| Personal Web Sever | PWS |
| Simple API for XML | SAX |
| Standard Generalized Markup Language | SGML |
| Subscribe & Receive Data Distribution Mechanism | SRDDM |

THIS PAGE INTENTIONALLY LEFT BLANK

## APPENDIX E (XML DOCUMENT)

```
<?xml version="1.0" encoding="UTF-8"?>
<Target_Mission>
        <Mission_ID>152-XX-221</Mission_ID>
        <Operation_Name>Tandem Thrust</Operation_Name>
        <Classification_Level>T</Classification_Level>
        <Codeword>5</Codeword>
        <Mission_Name>Strike Package 322</Mission_Name>
        <Target_Objective>
                <Country>IQ</Country>
                <Execution_Date>20000927</Execution_Date>
                <Functional_Production_Area>FUELS</Functional_Production_Area>
                <Priority_Objective>3</Priority_Objective>
                <Record_Status>E</Record_Status>
                <Domain_Level>SI</Domain_Level>
                <Eval>2</Eval>
                <Originating_Agency>EA</Originating_Agency>
                <Objective_Name>Airfield in Area 301</Objective_Name>
        </Target_Objective>
        <Target_List>
                <Operation_Name>Tandem Thrust</Operation_Name>
                <Classification_Level>T</Classification_Level>
                <Date_Created>20000825194500</Date_Created>
                <Date_Last_Changed>00000000000000</Date_Last_Changed>
                <Domain_Level>SI</Domain_Level>
                <Target_List_ID>12226</Target_List_ID>
                <Target_List_Status>A</Target_List_Status>
                <Target_List_Type>JTL</Target_List_Type>
                <Target_List_Name>Area 301 Tamino Airfield
Desig</Target_List_Name>
                <Production_Level>S</Production_Level>
                <Record_Status>E</Record_Status>
                <Target>
                        <Affiliation>H</Affiliation>
                        <Country>IQ</Country>
                        <Classification_Level>T</Classification_Level>
                        <Condition>COM</Condition>
                        <Coordinates>325218290N1170928640E</Coordinates>
                        <Coordinate_Basis>2</Coordinate_Basis>
                        <Coordinate_Derivative>PM</Coordinate_Derivative>
                        <Date_Created>19991011160000</Date_Created>
                        <Date_Last_Change>00000000000000</Date_Last_Change>
                        <Hardness>M</Hardness>
                        <Height>320.0</Height>
                        <Domain_Level>SI</Domain_Level>
```

<Elevation>2040</Elevation>
<Elevation_Confidence>100</Elevation_Confidence>
<Target_Name>Control Tower</Target_Name>
<Evaluation>1</Evaluation>
<Radius>125.0</Radius>
<Review_Date>20000825190000</Review_Date>
<Release_Mark>MQ</Release_Mark>
<Facility>
<Access>CLRMO</Access>
<Activity>ATC</Activity>
<BE_Number>1014-8Z-3967</BE_Number>
<Category>40812</Category>
<Evaluation>1</Evaluation>
<Facility_Name>Tamino Control Tower</Facility_Name>
<Facility_ID>32008</Facility_ID>
<Location_Name>Tamino Airfield</Location_Name>
<Primary_Mission>DQ</Primary_Mission>
<Relative_Ranking>1</Relative_Ranking>

<Population_Area_Proximity>14</Population_Area_Proximity>
<Record_Status>E</Record_Status>
<Review_Date>20000825190000</Review_Date>
<Graphic_Agency>DIA</Graphic_Agency>
<Graphic_Country>US</Graphic_Country>
</Facility>
</Target>
<Target>
<Affiliation>H</Affiliation>
<Country>IQ</Country>
<Classification_Level>T</Classification_Level>
<Condition>COM</Condition>
<Coordinates>325177560N1170823930E</Coordinates>
<Coordinate_Basis>2</Coordinate_Basis>
<Coordinate_Derivative>PM</Coordinate_Derivative>
<Date_Created>19991011160000</Date_Created>
<Date_Last_Change>00000000000000</Date_Last_Change>
<Hardness>H</Hardness>
<Height>20.0</Height>
<Domain_Level>SI</Domain_Level>
<Elevation>2040</Elevation>
<Elevation_Confidence>100</Elevation_Confidence>
<Target_Name>Bunker</Target_Name>
<Evaluation>1</Evaluation>
<Radius>235.0</Radius>
<Review_Date>20000825190000</Review_Date>
<Release_Mark>MQ</Release_Mark>

<Facility>
<Access>CLRMO</Access>
<Activity>STG</Activity>
<BE_Number>1014-8Z-3976</BE_Number>
<Category>40812</Category>
<Evaluation>1</Evaluation>
<Facility_Name>Bunker for A/C Storage</Facility_Name>
<Facility_ID>32010</Facility_ID>
<Location_Name>Tamino Airfield</Location_Name>
<Primary_Mission>DQ</Primary_Mission>
<Relative_Ranking>2</Relative_Ranking>

<Population_Area_Proximity>14</Population_Area_Proximity>
<Record_Status>E</Record_Status>
<Review_Date>20000825190000</Review_Date>
<Graphic_Agency>DIA</Graphic_Agency>
<Graphic_Country>US</Graphic_Country>
</Facility>
</Target>
<Target>
<Affiliation>H</Affiliation>
<Country>IQ</Country>
<Classification_Level>T</Classification_Level>
<Condition>COM</Condition>
<Coordinates>325377680N1171033970E</Coordinates>
<Coordinate_Basis>2</Coordinate_Basis>
<Coordinate_Derivative>PM</Coordinate_Derivative>
<Date_Created>19991011160000</Date_Created>
<Date_Last_Change>19991205132000</Date_Last_Change>
<Hardness>H</Hardness>
<Height>0.0</Height>
<Domain_Level>SI</Domain_Level>
<Elevation>2040</Elevation>
<Elevation_Confidence>100</Elevation_Confidence>
<Target_Name>Runway</Target_Name>
<Evaluation>1</Evaluation>
<Radius>2000.0</Radius>
<Review_Date>20000825190000</Review_Date>
<Release_Mark>MQ</Release_Mark>
</Target>
</Target_List>
</Target_Mission>

THIS PAGE INTENTIONALLY LEFT BLANK

## APPENDIX F (XML DOCUMENT WITH DTD)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!DOCTYPE Target_Mission [
<!ELEMENT Access (#PCDATA)>
<!ATTLIST Activity
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "3">
<!ELEMENT Activity (#PCDATA)>
<!ATTLIST Affiliation
        dtype NMTOKEN #FIXED "char">
<!ELEMENT Affiliation (#PCDATA)>
<!ATTLIST BE_Number
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "10">
<!ELEMENT BE_Number (#PCDATA)>
<!ATTLIST Category
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "5">
<!ELEMENT Category (#PCDATA)>
<!ATTLIST Classification_Level
        dtype NMTOKEN #FIXED "char">
<!ELEMENT Classification_Level (#PCDATA)>
<!ATTLIST Codeword
        dtype NMTOKEN #FIXED "char">
<!ELEMENT Codeword (#PCDATA)>
<!ATTLIST Condition
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "4">
<!ELEMENT Condition (#PCDATA)>
<!ATTLIST Coordinate_Basis
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "2">
<!ELEMENT Coordinate_Basis (#PCDATA)>
<!ATTLIST Coordinate_Derivative
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "2">
<!ELEMENT Coordinate_Derivative (#PCDATA)>
<!ATTLIST Coordinates
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "21">
<!ELEMENT Coordinates (#PCDATA)>
<!ATTLIST Country
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "2">
```

```
<!ELEMENT Country (#PCDATA)>
<!ATTLIST Date_Created
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "14">
<!ELEMENT Date_Created (#PCDATA)>
<!ATTLIST Date_Last_Change
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "14">
<!ELEMENT Date_Last_Change (#PCDATA)>
<!ATTLIST Domain_Level
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "2">
<!ELEMENT Domain_Level (#PCDATA)>
<!ATTLIST Elevation
        dtype NMTOKEN #FIXED "float">
<!ELEMENT Elevation (#PCDATA)>
<!ELEMENT Elevation_Confidence (#PCDATA)>
<!ATTLIST Eval
        dtype NMTOKEN #FIXED "char">
<!ELEMENT Eval (#PCDATA)>
<!ATTLIST Evaluation
        dtype NMTOKEN #FIXED "char">
<!ELEMENT Evaluation (#PCDATA)>
<!ATTLIST Execution_Date
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "8">
<!ELEMENT Execution_Date (#PCDATA)>
<!ELEMENT Facility (Access, Activity, BE_Number, Category, Evaluation,
Facility_Name, Facility_ID, Location_Name?, Primary_Mission?, Relative_Ranking?,
Population_Area_Proximity?, Record_Status, Review_Date, Graphic_Agency,
Graphic_Country)>
<!ATTLIST Facility_ID
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "14">
<!ELEMENT Facility_ID (#PCDATA)>
<!ATTLIST Facility_Name
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "54">
<!ELEMENT Facility_Name (#PCDATA)>
<!ATTLIST Functional_Production_Area
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "5">
<!ELEMENT Functional_Production_Area (#PCDATA)>
<!ATTLIST Graphic_Agency
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "15">
```

```
<!ELEMENT Graphic_Agency (#PCDATA)>
<!ATTLIST Graphic_Country
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "2">
<!ELEMENT Graphic_Country (#PCDATA)>
<!ATTLIST Hardness
        dtype NMTOKEN #FIXED "char">
<!ELEMENT Hardness (#PCDATA)>
<!ATTLIST Height
        dtype NMTOKEN #FIXED "float">
<!ELEMENT Height (#PCDATA)>
<!ATTLIST Location_Name
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "54">
<!ELEMENT Location_Name (#PCDATA)>
<!ELEMENT Mission_ID (#PCDATA)>
<!ATTLIST Mission_ID
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "15">
<!ATTLIST Mission_Name
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "30">
<!ELEMENT Mission_Name (#PCDATA)>
<!ATTLIST Objective_Name
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "54">
<!ELEMENT Objective_Name (#PCDATA)>
<!ATTLIST Operation_Name
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "54">
<!ELEMENT Operation_Name (#PCDATA)>
<!ELEMENT Originating_Agency (#PCDATA)>
<!ATTLIST Population_Area_Proximity
        dtype NMTOKEN #FIXED "char">
<!ELEMENT Population_Area_Proximity (#PCDATA)>
<!ATTLIST Primary_Mission
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "4">
<!ELEMENT Primary_Mission (#PCDATA)>
<!ATTLIST Priority_Objective
        dtype NMTOKEN #FIXED "smallint">
<!ELEMENT Priority_Objective (#PCDATA)>
<!ATTLIST Production_Level
        dtype NMTOKEN #FIXED "char">
<!ELEMENT Production_Level (#PCDATA)>
<!ATTLIST Radius
```

```
        dtype NMTOKEN #FIXED "float">
<!ELEMENT Radius (#PCDATA)>
<!ATTLIST Record_Status
        dtype NMTOKEN #FIXED "char">
<!ELEMENT Record_Status (#PCDATA)>
<!ATTLIST Relative_Ranking
        dtype NMTOKEN #FIXED "int">
<!ELEMENT Relative_Ranking (#PCDATA)>
<!ATTLIST Release_Mark
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "2">
<!ELEMENT Release_Mark (#PCDATA)>
<!ATTLIST Review_Date
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "14">
<!ELEMENT Review_Date (#PCDATA)>
<!ELEMENT Target (Affiliation?, Country?, Classification_Level, Condition,
Coordinates, Coordinate_Basis, Coordinate_Derivative, Date_Created,
Date_Last_Change, Hardness?, Height?, Domain_Level, Elevation?,
Elevation_Confidence, Target_Name, Evaluation, Radius?, Review_Date,
Release_Mark?, Facility?)>
<!ELEMENT Target_List (Operation_Name?, Classification_Level, Date_Created,
Date_Last_Change, Domain_Level, Target_List_ID, Target_List_Status,
Target_List_Type, Target_List_Name, Production_Level, Record_Status, Target+)>
<!ATTLIST Target_List_ID
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "14">
<!ELEMENT Target_List_ID (#PCDATA)>
<!ATTLIST Target_List_Name
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "54">
<!ELEMENT Target_List_Name (#PCDATA)>
<!ATTLIST Target_List_Status
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "3">
<!ELEMENT Target_List_Status (#PCDATA)>
<!ATTLIST Target_List_Type
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "3">
<!ELEMENT Target_List_Type (#PCDATA)>
<!ELEMENT Target_Mission (Mission_ID?, Operation_Name?, Classification_Level,
Codeword?, Mission_Name, Target_Objective, Target_List)>
<!ATTLIST Target_Name
        dtype NMTOKEN #FIXED "string"
        dsize NMTOKEN #FIXED "54">
<!ELEMENT Target_Name (#PCDATA)>
```

```
<!ELEMENT Target_Objective (Country?, Execution_Date?,
Functional_Production_Area?, Priority_Objective?, Record_Status, Domain_Level, Eval,
Originating_Agency, Objective_Name)>
]>


<Target_Mission>
        <Mission_ID>152-XX-221</Mission_ID>
        <Operation_Name>Tandem Thrust</Operation_Name>
        <Classification_Level>T</Classification_Level>
        <Codeword>5</Codeword>
        <Mission_Name>Strike Package 322</Mission_Name>
        <Target_Objective>
                <Country>IQ</Country>
                <Execution_Date>20000927</Execution_Date>
                <Functional_Production_Area>FUELS</Functional_Production_Area>
                <Priority_Objective>3</Priority_Objective>
                <Record_Status>E</Record_Status>
                <Domain_Level>SI</Domain_Level>
                <Eval>2</Eval>
                <Originating_Agency>EA</Originating_Agency>
                <Objective_Name>Airfield in Area 301</Objective_Name>
        </Target_Objective>
        <Target_List>
                <Operation_Name>Tandem Thrust</Operation_Name>
                <Classification_Level>T</Classification_Level>
                <Date_Created>20000825194500</Date_Created>
                <Date_Last_Changed>00000000000000</Date_Last_Changed>
                <Domain_Level>SI</Domain_Level>
                <Target_List_ID>12226</Target_List_ID>
                <Target_List_Status>A</Target_List_Status>
                <Target_List_Type>JTL</Target_List_Type>
                <Target_List_Name>Area 301 Tamino Airfield
Desig</Target_List_Name>
                <Production_Level>S</Production_Level>
                <Record_Status>E</Record_Status>
                <Target>
                        <Affiliation>H</Affiliation>
                        <Country>IQ</Country>
                        <Classification_Level>T</Classification_Level>
                        <Condition>COM</Condition>
                        <Coordinates>325218290N1170928640E</Coordinates>
                        <Coordinate_Basis>2</Coordinate_Basis>
                        <Coordinate_Derivative>PM</Coordinate_Derivative>
                        <Date_Created>19991011160000</Date_Created>
                        <Date_Last_Change>00000000000000</Date_Last_Change>
```

<Hardness>M</Hardness>
<Height>320.0</Height>
<Domain_Level>SI</Domain_Level>
<Elevation>2040</Elevation>
<Elevation_Confidence>100</Elevation_Confidence>
<Target_Name>Control Tower</Target_Name>
<Evaluation>1</Evaluation>
<Radius>125.0</Radius>
<Review_Date>20000825190000</Review_Date>
<Release_Mark>MQ</Release_Mark>
<Facility>
<Access>CLRMO</Access>
<Activity>ATC</Activity>
<BE_Number>1014-8Z-3967</BE_Number>
<Category>40812</Category>
<Evaluation>1</Evaluation>
<Facility_Name>Tamino Control Tower</Facility_Name>
<Facility_ID>32008</Facility_ID>
<Location_Name>Tamino Airfield</Location_Name>
<Primary_Mission>DQ</Primary_Mission>
<Relative_Ranking>1</Relative_Ranking>

<Population_Area_Proximity>14</Population_Area_Proximity>
<Record_Status>E</Record_Status>
<Review_Date>20000825190000</Review_Date>
<Graphic_Agency>DIA</Graphic_Agency>
<Graphic_Country>US</Graphic_Country>
</Facility>
</Target>
<Target>
<Affiliation>H</Affiliation>
<Country>IQ</Country>
<Classification_Level>T</Classification_Level>
<Condition>COM</Condition>
<Coordinates>325177560N1170823930E</Coordinates>
<Coordinate_Basis>2</Coordinate_Basis>
<Coordinate_Derivative>PM</Coordinate_Derivative>
<Date_Created>19991011160000</Date_Created>
<Date_Last_Change>00000000000000</Date_Last_Change>
<Hardness>H</Hardness>
<Height>20.0</Height>
<Domain_Level>SI</Domain_Level>
<Elevation>2040</Elevation>
<Elevation_Confidence>100</Elevation_Confidence>
<Target_Name>Bunker</Target_Name>
<Evaluation>1</Evaluation>

<Radius>235.0</Radius>
<Review_Date>20000825190000</Review_Date>
<Release_Mark>MQ</Release_Mark>
<Facility>
<Access>CLRMO</Access>
<Activity>STG</Activity>
<BE_Number>1014-8Z-3976</BE_Number>
<Category>40812</Category>
<Evaluation>1</Evaluation>
<Facility_Name>Bunker for A/C Storage</Facility_Name>
<Facility_ID>32010</Facility_ID>
<Location_Name>Tamino Airfield</Location_Name>
<Primary_Mission>DQ</Primary_Mission>
<Relative_Ranking>2</Relative_Ranking>
<Population_Area_Proximity>14</Population_Area_Proximity>
<Record_Status>E</Record_Status>
<Review_Date>20000825190000</Review_Date>
<Graphic_Agency>DIA</Graphic_Agency>
<Graphic_Country>US</Graphic_Country>
</Facility>
</Target>
<Target>
<Affiliation>H</Affiliation>
<Country>IQ</Country>
<Classification_Level>T</Classification_Level>
<Condition>COM</Condition>
<Coordinates>325377680N1171033970E</Coordinates>
<Coordinate_Basis>2</Coordinate_Basis>
<Coordinate_Derivative>PM</Coordinate_Derivative>
<Date_Created>19991011160000</Date_Created>
<Date_Last_Change>19991205132000</Date_Last_Change>
<Hardness>H</Hardness>
<Height>0.0</Height>
<Domain_Level>SI</Domain_Level>
<Elevation>2040</Elevation>
<Elevation_Confidence>100</Elevation_Confidence>
<Target_Name>Runway</Target_Name>
<Evaluation>1</Evaluation>
<Radius>2000.0</Radius>
<Review_Date>20000825190000</Review_Date>
<Release_Mark>MQ</Release_Mark>
</Target>
</Target_List>
</Target_Mission>

THIS PAGE INTENTIONALLY LEFT BLANK

## APPENDIX G (PHP4 API FOR XML EXAMPLE PROGRAM)

```
<?php
if( ! ($fp = fopen("./target_info.xml","r")))
  die("Couldn't open xml!");
  $target_counter = 0;
  $target_data = array();
  $xml_current_tag_state = '';



function startElementHandler( $parser, $element_name, $element_attribs )
{
        global $target_counter;
        global $target_data;
        global $xml_current_tag_state;

        $xml_current_tag_state = $element_name;
}



function endElementHandler( $parser, $element_name )
{
        global $target_counter;
        global $target_data;
        global $xml_current_tag_state;

        $xml_current_tag_state = '';


        if( $element_name == "TARGET" )
        {
                $target_counter++;
        }
}



function characterDataHandler( $parser, $data )
{
        global $target_counter;
        global $target_data;
```

```
global $xml_current_tag_state;

if( $xml_current_tag_state == '')
 return;

if( $xml_current_tag_state == "AFFILIATION" ) {
 $target_data[$target_counter]["Affiliation"] = $data;
}
if( $xml_current_tag_state == "COUNTRY" ) {
 $target_data[$target_counter]["Country"] = $data;
}
if( $xml_current_tag_state == "CLASSIFICATION_LEVEL" ) {
 $target_data[$target_counter]["Classification_Level"] = $data;
}
if( $xml_current_tag_state == "CONDITION" ) {
 $target_data[$target_counter]["Condition"] = $data;
}
if( $xml_current_tag_state == "COORDINATES" ) {
 $target_data[$target_counter]["Coordinates"] = $data;
}
if( $xml_current_tag_state == "COORDINATE_BASIS" ) {
 $target_data[$target_counter]["Coordinate_Basis"] = $data;
}
if( $xml_current_tag_state == "COORDINATE_DERIVATIVE" ) {
 $target_data[$target_counter]["Coordinate_Derivative"] = $data;
}
if( $xml_current_tag_state == "DATE_CREATED" ) {
 $target_data[$target_counter]["Date_Created"] = $data;
}
if( $xml_current_tag_state == "DATE_LAST_CHANGE" ) {
 $target_data[$target_counter]["Date_Last_Change"] = $data;
}
if( $xml_current_tag_state == "HARDNESS" ) {
 $target_data[$target_counter]["Hardness"] = $data;
}
if( $xml_current_tag_state == "HEIGHT" ) {
 $target_data[$target_counter]["Height"] = $data;
}
if( $xml_current_tag_state == "DOMAIN_LEVEL" ) {
 $target_data[$target_counter]["Domain_Level"] = $data;
}
if( $xml_current_tag_state == "ELEVATION" ) {
 $target_data[$target_counter]["Elevation"] = $data;
}
if( $xml_current_tag_state == "ELEVATION_CONFIDENCE" ) {
 $target_data[$target_counter]["Elevation_Confidence"] = $data;
```

```
 }
 if( $xml_current_tag_state == "TARGET_NAME" ) {
  $target_data[$target_counter]["Target_Name"] = $data;
 }
 if( $xml_current_tag_state == "EVALUATION" ) {
  $target_data[$target_counter]["Evaluation"] = $data;
 }
 if( $xml_current_tag_state == "RADIUS" ) {
  $target_data[$target_counter]["Radius"] = $data;
 }
 if( $xml_current_tag_state == "REVIEW_DATE" ) {
  $target_data[$target_counter]["Review_Date"] = $data;
 }
 if( $xml_current_tag_state == "RELEASE_MARK" ) {
  $target_data[$target_counter]["Release_Mark"] = $data;
 }
 if( $xml_current_tag_state == "ACCESS" ) {
  $target_data[$target_counter]["Access"] = $data;
 }
 if( $xml_current_tag_state == "ACTIVITY" ) {
  $target_data[$target_counter]["Activity"] = $data;
 }
 if( $xml_current_tag_state == "BE_NUMBER" ) {
  $target_data[$target_counter]["BE_Number"] = $data;
 }
 if( $xml_current_tag_state == "CATEGORY" ) {
  $target_data[$target_counter]["Category"] = $data;
 }
 if( $xml_current_tag_state == "EVALUATION" ) {
  $target_data[$target_counter]["Evaluation"] = $data;
 }
 if( $xml_current_tag_state == "FACILITY_NAME" ) {
  $target_data[$target_counter]["Facility_Name"] = $data;
 }
 if( $xml_current_tag_state == "FACILITY_ID" ) {
  $target_data[$target_counter]["Facility_ID"] = $data;
 }
 if( $xml_current_tag_state == "LOCATION_NAME" ) {
  $target_data[$target_counter]["Location_Name"] = $data;
 }
 if( $xml_current_tag_state == "PRIMARY_MISSION" ) {
  $target_data[$target_counter]["Primary_Mission"] = $data;
 }
 if( $xml_current_tag_state == "RELATIVE_RANKING" ) {
  $target_data[$target_counter]["Relative_Ranking"] = $data;
 }
```

```
        if( $xml_current_tag_state == "POPULATION_AREA_PROXIMITY" ) {
          $target_data[$target_counter]["Population_Area_Proximity"] = $data;
        }
        if( $xml_current_tag_state == "RECORD_STATUS" ) {
          $target_data[$target_counter]["Record_Status"] = $data;
        }
        if( $xml_current_tag_state == "REVIEW_DATE" ) {
          $target_data[$target_counter]["Review_Date"] = $data;
        }
        if( $xml_current_tag_state == "GRAPHIC_AGENCY" ) {
          $target_data[$target_counter]["Graphic_Agency"] = $data;
        }
        if( $xml_current_tag_state == "GRAPHIC_COUNTRY" ) {
          $target_data[$target_counter]["Graphic_Country"] = $data;
        }

}
if( !($xml_parser = xml_parser_create()) )
  die("Couldn't create XML parser!");
xml_set_element_handler($xml_parser, "startElementHandler", "endElementHandler");
xml_set_character_data_handler($xml_parser, "characterDataHandler");


while( $data = fread($fp, 4096) )
{
 if( !xml_parse($xml_parser, $data, feof($fp)) )
   {
        break; // get out of while loop if we're done with the file
   }
}
xml_parser_free($xml_parser);?>



<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN">
<HTML>
<HEAD>
        <TITLE> Parsing the Sample XML File</TITLE>
</HEAD>
<BODY BGCOLOR="#ffffff">


<?php
echo "Target Mission: " . "<BR>\n";
for( $i=0 ; $i < $target_counter ; ++$i)
{
```

```
        $font_color = "#ff0000";
        echo "<FONT SIZE=\"+1\" color=\"$font_color\">" .
            $target_data[$i]["Target_Mission"] . "</FONT><BR>\n";
        echo "   Target: " . $i . "<BR>\n";
        echo "     Affiliation:   " .
$target_data[$i]["Affiliation"] . "<BR>\n";
        echo "     Country:   " .
$target_data[$i]["Country"] . "<BR>\n";
        echo "     Classification Level:   " .
$target_data[$i]["Classification_Level"] . "<BR>\n";
        echo "     Condition:   " .
$target_data[$i]["Condition"] . "<BR>\n";
        echo "     Coordinates:   " .
$target_data[$i]["Coordinates"] . "<BR>\n";
        echo "     Coordinate Basis:   " .
$target_data[$i]["Coordinate_Basis"] . "<BR>\n";
        echo "     Coordinate Derivative:   " .
$target_data[$i]["Coordinate_Derivative"] . "<BR>\n";
        echo "     Date Created:   " .
$target_data[$i]["Date_Created"] . "<BR>\n";
        echo "     Date Last Change:   " .
$target_data[$i]["Date_Last_Change"] . "<BR>\n";
        echo "     Hardness:   " .
$target_data[$i]["Hardness"] . "<BR>\n";
        echo "     Height:   " .
$target_data[$i]["Height"] . "<BR>\n";
        echo "     Domain Level:   " .
$target_data[$i]["Domain_Level"] . "<BR>\n";
        echo "     Elevation:   " .
$target_data[$i]["Elevation"] . "<BR>\n";
        echo "     Elevation Confidence:   " .
$target_data[$i]["Elevation_Confidence"] . "<BR>\n";
        echo "     Target Name:   " .
$target_data[$i]["Target_Name"] . "<BR>\n";
        echo "     Evaluation:   " .
$target_data[$i]["Evaluation"] . "<BR>\n";
        echo "     Radius:   " .
$target_data[$i]["Radius"] . "<BR>\n";
        echo "     Review Date:   " .
$target_data[$i]["Review_Date"] . "<BR>\n";
        echo "     Release Mark:   " .
$target_data[$i]["Release_Mark"] . "<BR>\n";
        echo "     Facility Info: " . "<BR>\n";
        echo "        Access:   " .
$target_data[$i]["Access"] . "<BR>\n";
```

```
        echo "        Activity:   " .
$target_data[$i]["Activity"] . "<BR>\n";
        echo "        BE Number:   " .
$target_data[$i]["BE_Number"] . "<BR>\n";
        echo "        Category:   " .
$target_data[$i]["Category"] . "<BR>\n";
        echo "        Evaluation:   " .
$target_data[$i]["Evaluation"] . "<BR>\n";
        echo "        Facility Name:
  " . $target_data[$i]["Facility_Name"] . "<BR>\n";
        echo "        Facility ID:   " .
$target_data[$i]["Facility_ID"] . "<BR>\n";
        echo "        Location Name:
  " . $target_data[$i]["Location_Name"] . "<BR>\n";
        echo "        Primary Mission:
  " . $target_data[$i]["Primary_Mission"] . "<BR>\n";
        echo "        Relative Ranking:
  " . $target_data[$i]["Relative_Ranking"] . "<BR>\n";
        echo "        Population Area Proximity:
  " . $target_data[$i]["Population_Area_Proximity"] . "<BR>\n";
        echo "        Record Status:
  " . $target_data[$i]["Record_Status"] . "<BR>\n";
        echo "        Review Date:   "
. $target_data[$i]["Review_Date"] . "<BR>\n";
        echo "        Graphic Agency:
  " . $target_data[$i]["Graphic_Agency"] . "<BR>\n";
        echo "        Graphic Country:
  " . $target_data[$i]["Graphic_Country"] . "<BR>\n";

        echo "<BR>\n";
}
?>
</BODY>
                                        </HTML>
```

## APPENDIX H (WEB BROWSER OUTPUT OF XML DOCUMENT)

Target Mission:

Target: 0
Affiliation: H
Country: IQ
Classification Level: T
Condition: COM
Coordinates: 325218290N1170928640E
Coordinate Basis: 2
Coordinate Derivative: PM
Date Created: 19991011160000
Date Last Change: 00000000000000
Hardness: M
Height: 320.0
Domain Level: SI
Elevation: 2040
Elevation Confidence: 100
Target Name: Control Tower
Evaluation: 1
Radius: 125.0
Review Date: 20000825190000
Release Mark: MQ
Facility Info:
Access: CLRMO
Activity: ATC
BE Number: 1014-8Z-3967
Category: 40812
Evaluation: 1
Facility Name: Tamino Control Tower
Facility ID: 32008
Location Name: Tamino Airfield
Primary Mission: DQ
Relative Ranking: 1
Population Area Proximity: 14
Record Status: E
Review Date: 20000825190000
Graphic Agency: DIA
Graphic Country: US

Target: 1
Affiliation: H
Country: IQ
Classification Level: T

Condition: COM
Coordinates: 325177560N1170823930E
Coordinate Basis: 2
Coordinate Derivative: PM
Date Created: 19991011160000
Date Last Change: 00000000000000
Hardness: H
Height: 20.0
Domain Level: SI
Elevation: 2040
Elevation Confidence: 100
Target Name: Bunker
Evaluation: 1
Radius: 235.0
Review Date: 20000825190000
Release Mark: MQ
Facility Info:
Access: CLRMO
Activity: STG
BE Number: 1014-8Z-3976
Category: 40812
Evaluation: 1
Facility Name: Bunker for A/C Storage
Facility ID: 32010
Location Name: Tamino Airfield
Primary Mission: DQ
Relative Ranking: 2
Population Area Proximity: 14
Record Status: E
Review Date: 20000825190000
Graphic Agency: DIA
Graphic Country: US

Target: 2
Affiliation: H
Country: IQ
Classification Level: T
Condition: COM
Coordinates: 325377680N1171033970E
Coordinate Basis: 2
Coordinate Derivative: PM
Date Created: 19991011160000
Date Last Change: 19991205132000
Hardness: H
Height: 0.0
Domain Level: SI
Elevation: 2040
Elevation Confidence: 100
Target Name: Runway
Evaluation: 1
Radius: 2000.0
Review Date: 20000825190000
Release Mark: MQ
Facility Info:
Access:
Activity:
BE Number:
Category:
Evaluation: 1
Facility Name:
Facility ID:
Location Name:
Primary Mission:
Relative Ranking:
Population Area Proximity:
Record Status:
Review Date: 20000825190000
Graphic Agency:
Graphic Country:

THIS PAGE INTENTIONALLY LEFT BLANK

## LIST OF REFERENCES


[PHPSITE] "Manual: PHP Development" , http://www.php.net/distributions/bigmanual.html

[XML] "Extensible Markup Language 1.0" W3C Recommendation, http://www.w3.org/TR/REC-xml, February 1998.

[MASXML] Navarro A. and White C., "Mastering XML", 2000.

[PHP400] Choi W., Kent A., Lea C., Parsad G., and Ullman, "Beginning PHP4", 2000.

[PHPNET] Calo S., "PHP Builder", http://phpbuilder.com/tips/item.php.

[INFSITE] "Informix XML Strategy: An Overview", http://www.informix.com/xml.

[INFXML] Brown P., "Informix And XML". http://www.gca.org/papers/xmleurope2000/papers/s35-04.html.

[INFWDB] "Informix Web DataBlade Module, Version 4.0 for Unix and NT", http://www.informix.com/informix/techbriefs/iif2000/module.pdf.

[NRC] "Realizing the Potential of C4I: Fundamental Challenges", Committee to Review DOD C4I Plans and Programs, National Research Council, 1999.

[REN] Renner, S. and Scarano, J., "Migrating Legacy Applications to a Shared Data Environment", Proceedings of the Federal Database Colloquium and Exposition, August 1996.

[HINA] Hina, D. "Evaluation of the Extensible Markup Language (XML) As A Means For Establishing Interoperability Between Heterogeneous Department of Defense (DOD) Databases".

[DII COE1] "DII COE Namespaces", http://diides.ncr.disa.mil/xmlreg.namespace_list.cfm.

[SAX] http://www.megginson.com/SAX.

THIS PAGE INTENTIONALLY LEFT BLANK

## INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center ..... 2
   8725 John J. Kingman Rd., STE 0944
   Ft. Belvoir, Virginia 22060-6218

2. Dudley Knox Library ..... 2
   Naval Postgraduate School
   411 Dyer Road
   Monterey, California 93943-5101

3. Dr. Valdis Berzins, Code CS/Bz ..... 1
   Computer Science Department
   Naval Postgraduate School
   Monterey, California 93943-5100

4. Prof. Luqi, Code CS/Lq ..... 1
   Computer Science Department
   Naval Postgraduate School
   Monterey, California 93943-5100

5. CAPT Paul Young ..... 1
   Computer Science Department
   Naval Postgraduate School
   Monterey, California 93943-5100

6. Chairman, Code CS ..... 1
   Naval Postgraduate School
   Monterey, CA 93943-5100

7. Mike Saboe, Associate Director ..... 1
   U.S. Army Next Generation Software Engineering
   ATTN: AMSTA-TR-R/265
   Warren, MI 48397-5000

8. Eddie L. Davis ..... 3
   23710 Radclift
   Oak Park, MI 48237